ATID Application Development Framework Reference Manual–Barcode Scanner

Revision: Ver. 0.4

Date: January, 2014

ATID Co.,Ltd

Table of Contents

Acronym

modules	descriptions
AADF	ATIDApplication Development Framework

Revision History

Version	Date	Reason	Description	Author
0.1	2012/01/17	Draft		Y. J. CHO
0.2	2013/01/25	Update	1. Symbology Configuration API added. 2. Content related to RAW Data added.	Y. J. CHO
0.3	2013/02/26	Update	1. Function of LED Mode added.	Y. J. CHO
0.4	2014/01/10	Update	1. Illumination Power level Control Function added.	Y. J. CHO

1 .NET API Reference

1.1 Enumerations

1.1.1 BARCODE_RESULT

The result of a call to functions.

- **BARCODE_RESULT_ALREADY_BARCODE_SELECTED**
  Barcode device was already selected.
- **BARCODE_RESULT_ALREADY_OPENED**
  Barcode device was already opened.
- **BARCODE_RESULT_BARCODE_INIT_FAILURE**
  Barcode device initialization failed
- **BARCODE_RESULT_CAM_SELECTED**
  CAM device was already selected
- **BARCODE_RESULT_FAILURE**
  Function execution failed
- **BARCODE_RESULT_INVALID_ARGS**
  Invalid paramerter.
- **BARCODE_RESULT_INVALID_DEVICE**
  There is no barcode device.
- **BARCODE_RESULT_NOT_OPENED**
  Calling function without open
- **BARCODE_RESULT_OUTOFMEMORY**
  Failed to assign memory
- **BARCODE_RESULT_SUCCESS**
  Function execution success.
- **BARCODE_RESULT_UNSUPPORTED**
  Unsupported command.

1.1.2 IMAGE_SIZE

The size of image that will be captured through 2D Barcode Scanner.

- **CAP_IMG_PREVIEW**
  240 * 180
- **CAP_IMG_QVGA**
  320 * 240
- **CAP_IMG_VGA**

640 * 480

1.1.3 IMAGE_FORMAT

The format of image that will be captured through 2D Barcode Scanner.

- **IMG_FORMAT_JPEG**
  JPEG format
- **IMG_FORMAT_BMP**
  BMP format

1.1.4 SYMBOLOGIES_1D

Symbology ID can be decoded through 1D Barcode Scanner.

- **NUM_OF_1D_SYMBOLOGIES**
- **SYMBOL_1D_BOOKLAND**
- **SYMBOL_1D_CHINESE25**
- **SYMBOL_1D_CODABAR**
- **SYMBOL_1D_CODE11**
- **SYMBOL_1D_CODE128**
- **SYMBOL_1D_CODE39**
- **SYMBOL_1D_CODE93**
- **SYMBOL_1D_DISCRETE25**
- **SYMBOL_1D_EAN13**
- **SYMBOL_1D_EAN8**
- **SYMBOL_1D_INTERLEAVED25**
- **SYMBOL_1D_ISBT128**
- **SYMBOL_1D_MSI**
- **SYMBOL_1D_RSS14**
- **SYMBOL_1D_RSSEXPANDED**
- **SYMBOL_1D_RSSLIMITED**
- **SYMBOL_1D_TRIOPTIC39**
- **SYMBOL_1D_UCCEAN128**
- **SYMBOL_1D_UPCA**
- **SYMBOL_1D_UPCE**
- **SYMBOL_1D_UPCE1**

1.1.5 SYMBOLOGIES_2DSWD

Symbology ID can be decoded through 2D Barcode Scanner.

- **NUM_OF_2DSWD_SYMBOLOGIES**
- **SYMBOL_2DSWD_AUSPOST**
- **SYMBOL_2DSWD_AZTEC**
- **SYMBOL_2DSWD_BPO**
- **SYMBOL_2DSWD_CANPOST**
- **SYMBOL_2DSWD_CHINAPOST**
- **SYMBOL_2DSWD_CODABAR**
- **SYMBOL_2DSWD_CODABLOCK**
- **SYMBOL_2DSWD_CODE11**
- **SYMBOL_2DSWD_CODE128**
- **SYMBOL_2DSWD_CODE16K**
- **SYMBOL_2DSWD_CODE32**
- **SYMBOL_2DSWD_CODE39**
- **SYMBOL_2DSWD_CODE49**
- **SYMBOL_2DSWD_CODE93**
- **SYMBOL_2DSWD_COMPOSITE**
- **SYMBOL_2DSWD_COUPONCODE**
- **SYMBOL_2DSWD_DATAMATRIX**
- **SYMBOL_2DSWD_DUTCHPOST**
- **SYMBOL_2DSWD_EAN13**
- **SYMBOL_2DSWD_EAN8**
- **SYMBOL_2DSWD_GEN_CODE128**
- **SYMBOL_2DSWD_GS1_128**
- **SYMBOL_2DSWD_IATA25**
- **SYMBOL_2DSWD_IDTAG**
- **SYMBOL_2DSWD_INT25**
- **SYMBOL_2DSWD_ISBT**
- **SYMBOL_2DSWD_JAPOST**
- **SYMBOL_2DSWD_KOREAPOST**
- **SYMBOL_2DSWD_MATRIX25**
- **SYMBOL_2DSWD_MAXICODE**
- **SYMBOL_2DSWD_MESA**
- **SYMBOL_2DSWD_MICROPDF**
- **SYMBOL_2DSWD_MSI**
- **SYMBOL_2DSWD_OCR**
- **SYMBOL_2DSWD_PDF417**
- **SYMBOL_2DSWD_PLANET**
- **SYMBOL_2DSWD_PLESSEY**
- **SYMBOL_2DSWD_POSICODE**

- **SYMBOL_2DSWD_POSTNET**
- **SYMBOL_2DSWD_QR**
- **SYMBOL_2DSWD_RSS**
- **SYMBOL_2DSWD_STRT25**
- **SYMBOL_2DSWD_TELEPEN**
- **SYMBOL_2DSWD_TLCODE39**
- **SYMBOL_2DSWD_TRIOPTIC**
- **SYMBOL_2DSWD_UPCA**
- **SYMBOL_2DSWD_UPCE0**
- **SYMBOL_2DSWD_UPCE1**
- **SYMBOL_2DSWD_USPS4CB**

1.1.6 OCRDirection

This enumeration is ignored and no longer supported.

- **OCRDIRECTION_BottomToTop**
- **OCRDIRECTION_LeftToRight**
- **OCRDIRECTION_RightToLeft**
- **OCRDIRECTION_TopToBottom**

1.1.7 OCRMode

This determines which OCR fonts (if any) are selected for decoding.

- **OCRMODE_A**
- **OCRMODE_B**
- **OCRMODE_DISABLED**
- **OCRMODE_MICR_UNSUPPORTED**
- **OCRMODE_MONEY**

1.2 Structures

1.2.1 BARCODECAPTUREPARAMS

The structure that while capturing images through 2D Barcode Scanner.

```
Public structBARCODECAPTUREPARAMS
{
  ByteImgFormat
```

```
    ByteImgSize;
    StringsFilePath;
};
```

- **ImgFormat**
  format
- **ImgSize**
  size
- **sFilePath**
  the path of image storage(include file name).

1.2.2 GENERAL_CONFIG

The structure that while reading or setting up the security level, scanning angle of 1D Barcode Scanner.

```
Public structGENERAL_CONFIG
{
    boolbBiDirectionalRedundancy;
    intnAngle;
    intnLinearCodeSecurityLevel
}
```

- **bBiDirectionalRedundancy**
  Returns whether a bar code must be successfully scanned in both direction(forward and reverse) before being decoded.
- **nAngle**
  Scanning angle. 5=Narrow, 6=Wide
- **nLinearCodeSecurityLevel**
  Four levels(1~4) of decode security for linear code types are supported. Select a highger security level for low levels of bar code quality. As security levels increase, the decoder's aggressivenese decrease.

1.2.3 SYMBOL_CONFIG

The structure that contains setup information for setup of Symbology.

```
Public structSYMBOL_CONFIG
{
    IntPtrpConfigData;
    Int Symbol;
}
```

- **pConfigData**
  The pointer of structure of Symbology that change setup information.
- **Symbol**
  The ID of Symbology that change setup information

1.2.4 CONFIG_1D_CHINESE25

The structure that saves setup information of Chinese 2 of 5 symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CHINESE25
{
  boolbEnabled;
}
```

- **bEnabled**
  whether Chinese 2 of 5 is enabled or disabled

1.2.5 CONFIG_1D_CODABAR

The stricter that saves setup information of CodabarSymbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CODABAR
{
  boolbEnabled;
  bool bCLSIEditing;
  bool bNOTISEditing;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whetherCodabar is enabled or disabled
- **bCLSIEditing**
  whether CLSI Editing is enabled or disabled
- **bNOTISEditing**
  whether NOTIS Editing is enabled or disabled
- **nMaxLength**
  max length for Codabar
- **nMinLength**
  min length for Codabar

1.2.6 CONFIG_1D_CODE128

The structure that saves setup information of Code 128 Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CODE128
{
  boolbEnabled;
}
```

- **bEnabled**
  whether Code 128 is enabled or disabled

1.2.7 CONFIG_1D_CODE39

The structure that saves setup information of Code 39Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CODE39
{
  boolbEnabled;
  bool bCheckDigitVerify;
  bool bCode32Prefix;
  bool bConvertCode39ToCode32;
  bool bEnableTriopticCode39;
  bool bFullAscii;
  bool bXmitCheckDigit;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Code 39 is enabled or disabled
- **bCheckDigitVerify**
  Checks the integrity of a Code 39 symbol to ensure it complies with specified algorithms. Only Code 39 symbolswhich include a modulo 43 check digit are decoded.
- **bCode32Prefix**
  Appends the character 'A' to the start of the decode data if enabled.
- **bConvertCode39ToCode32**
  Converts Code 39 to Code 32.
- **bEnableTriopticCode39**
  whetherTrioptic Code 39 is enabled or disabled.
- **bFullAscii**
  Enables or disables Code 39 Full ASCII
- **bXmitCheckDigit**

whether Code 39 Check Digit is enabled or disabled.

- **nMaxLength**

  max length for Code 39

- **nMinLength**

  min length for Code 39

1.2.8 CONFIG_1D_CODE93

The structure that saves setup information of Code 93Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CODE93
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**

  whether Code 93 is enabled or disabled

- **nMaxLength**

  max length for Code 93

- **nMinLength**

  min length for Code 93

1.2.9 CONFIG_1D_DISCRET25

Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_DISCRET25
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**

  whether Discrete 2 of 5 is enabled or disabled

- **nMaxLength**

  max length for Discrete 2 of 5

- **nMinLength**

  min length for Discrete 2 of 5

1.2.10 CONFIG_1D_INTERLEAVED25

The structure that saves setup information of the Interleaved 2 of 5 Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_DISCRET25
{
  boolbEnabled;
  bool bConvertI2of5ToEAN13;
  bool bXmitCheckDigit;
  short nCheckDigitVerify;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whetherInverleaved 2 of 5 is enabled or disabled
- **bConvertI2of5ToEAN13**
  Returns whether a 14-character I 2 of 5 code is converted to EAN-13 and transmitted to the host as EAN-13.
- **bXmitCheckDigit**
  whetherInterleaved 2 of 5 Check Digit is enabled or disabled.
- **nCheckDigitVerify**
  Checks the integrity of an I 2 of 5 symbol to ensure it complies with specified algorithms, either USS (UniformSymbology Specification), or OPCC (Optical Product Code Council).
- **nMaxLength**
  max length for Inverleaved 2 of 5
- **nMinLength**
  min length for Inverleaved 2 of 5

1.2.11 CONFIG_1D_ISBT128

The structure that saves setup information of the ISBT 128 Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_ISBT128
{
  boolbEnabled;
}
```

- **bEnabled**
  whether ISBT 128 is enabled or disabled

1.2.12 CONFIG_1D_MSI

The structure that saves setup information of the MSISymbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_MSI
{
  boolbEnabled;
  bool bCheckDigitAlgorithm;
  bool bCheckDigitVerify;
  boolbXmitCheckDigit;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether MSI is enabled or disabled
- **bCheckDigitAlgorithm**
  When Two MSI check digits is selected, sets an additional verification is required to ensure integrity. Twofollowing algorithms may be selected: MOD10/MOD11 or MOD10/MOD10.
- **bCheckDigitVerify**
  Sets the number of check digits at the end of the bar code that verify the integrity of the data. At least one checkdigit is always required. Check digits are not automatically transmitted with the data.
- **bXmitCheckDigit**
  whether MSI Check Digit is transmitted with the data.
- **nMaxLength**
  max length for MSI
- **nMinLength**
  min length for MSI

1.2.13 CONFIG_1D_RSS

The structure that saves setup information of the RSSSymbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_MSI
{
  boolbRSS14Enabled;
  bool bRSSExpandedEnabled;
  bool bRSSLimitedEnabled;
  boolbConvertRSSToUPCEAN;
```

}

- **bRSS14Enabled**
  whether RSS 14 is enabled or disabled
- **bRSSExpandedEnabled**
  whether RSS Expanded is enabled or disabled
- **bRSSLimitedEnabled**
  whether RSS Limited is enabled or disabled
- **bConvertRSSToUPCEAN**
  convert to RSS to UPCEAN

1.2.14 CONFIG_1D_UPCEAN

The structure that saves setup information of the UPC/EAN Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_UPCEAN
{
  boolbBooklandEnabled;
  boolbConvertEAN8ToEAN13;
  boolbConvertUPCE1ToUPCA;
  boolbConvertUPCEToUPCA;
  boolbEAN13Eanbled;
  boolbEAN8Enabled;
  boolbEANZeroExtend;
  boolbUCCCouponExtendedCode;
  boolbUCCEAN128Enabled;
  boolbUPCAEnabled;
  boolbUPCE1Enabled;
  boolbUPCEEnabled;
  boolbXmitUPCACheckChar;
  boolbXmitUPCE1CheckChar;
  boolbXmitUPCECheckChar;
  shortnDecodeUPCEANSupplemental;
  shortnDecodeUPCEANSupplementalRedundancy;
  shortnUPCAPreamble;
  shortnUPCE1Preamble;
  shortnUPCEANSecurityLevel;
  shortnUPCEPreamble;
}
```

- **bBooklandEnabled**
  whetherBookland is enabled or disabled
- **bConvertEAN8ToEAN13**
  When EAN Zero Extend is enabled, labels the extended symbol as either an EAN-13 bar code, or an EAN-8 barcode.
  When EAN Zero Extend is disabled, this parameter has no effect on bar code data.
- **bConvertUPCE1ToUPCA**
  Converts UPC-E1 (zero suppressed) decoded data to UPC-A format before transmission. After conversion, datafollows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit).
- **bConvertUPCEToUPCA**
  Converts UPC-E (zero suppressed) decoded data to UPC-A format before transmission. After conversion, datafollows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit).
- **bEAN13Eanbled**
  whether EAN-13 is enabled or disabled
- **bEAN8Enabled**
  whether EAN-8 is enabled or disabled
- **bEANZeroExtend**
  When enabled, five leading zeros are added to decoded EAN-8 symbols to make them compatible in format toEAN-13 symbols.
- **bUCCCouponExtendedCode**
  whether UCC Coupon is enabled or disabled
- **bUCCEAN128Enabled**
  whether UCCEAN 128 is enabled or disabled
- **bUPCAEnabled**
  whether UPC-A is enabled or disabled
- **bUPCE1Enabled**
  whether UPC-E1 is enabled or disabled
- **bUPCEEnabled**
  whether UPC-E is enabled or disabled
- **bXmitUPCACheckChar**
  Transmits the symbol with or without the UPC-A check digit.
- **bXmitUPCE1CheckChar**
  Transmits the symbol with or without the UPC-E1 check digit.
- **bXmitUPCECheckChar**
  Transmits the symbol with or without the UPC-E check digit.
- **nDecodeUPCEANSupplemental**
  Sets Decode UPC/EAN Supplementals option.Supplementals are additionally

appended characters (2 or 5)according to specific code format conventions (e.g., UPC A+2, UPC E+2, EAN 8+2). Three options are available.

0 : Decode supplementals

1 : Ignore supplementals

2 : Auto discriminate supplementals

- **nDecodeUPCEANSupplementalRedundancy**

When Autodiscriminate UPC/EAN Supplementals selected, adjusts the number of times a symbol withoutsupplementals is decoded before transmission. The range is from 2 to 20 times. Five or above is recommendedwhen decoding a mix of UPC/EAN symbols with and without supplementals, and the autodiscriminate option isselected.

this is an integer value in the range.[2..20]

- **nUPCAPreamble**

Returns the selected UPC-A Preamble option: transmit system character only, transmit system character andcountry code ("0" for USA), or no preamble transmitted. The lead-in characters are considered part of the symbol.

0 : No preamble

1 : System character

2 : System character, country code

- **nUPCE1Preamble**

Returns the selected UPC-E1 Preamble option: transmit system character only, transmit system character andcountry code ("0" for USA), or no preamble transmitted. The lead-in characters are considered part of the symbol.

0 : No preamble

1 : System character

2 : System character, country code

- **nUPCEANSecurityLevel**

Sets the UPC/EAN Security Level. There are four levels of decode security for UPC/EAN bar codes. Select a higherlevel of security are provided for decreasing levels of bar code quality. There is an inverse relationship betweensecurity and decoder aggressiveness, so be sure to choose only that level of security necessary for any givenapplication.

- UPC/EAN Security Level 0: This default setting allows the decoder to operate in its most aggressive state,while providing sufficient security in decoding "in-spec" UPC/EAN bar codes.

- UPC/EAN Security Level 1: As bar code quality levels diminish, certain characters become prone tomisdecodes before others (i.e., 1, 2, 7, 8). If you are experiencing misdecodes of poorly printed bar codes,and the misdecodes are limited to these characters, select this security level.

- UPC/EAN Security Level 2: If you are experiencing misdecodes of poorly printed bar codes, and themisdecodes are not limited to characters 1, 2, 7, and 8, select this security level.

- UPCIEAN Security Level 3: If you have tried Security Level 2, and are still experiencing misdecodes, selectthis security level. Be advised, selecting this option is an extreme measure against mis-decoding severelyout of spec bar codes. Selection of this level of security significantly impairs the decoding ability of thedecoder. If this level of security is necessary you should try to improve the quality of your bar codes.

- **nUPCEPreamble**

Sets the UPC-E Preamble option. Three options are given for lead-in characters for UPC-E symbols transmitted tothe host device: transmit system character only, transmit system character and country code ("0" for USA), and nopreamble transmitted. The lead-in characters are considered part of the symbol.

0 : No preamble

1 : System character

2 : System character, country code

1.2.15 CONFIG_1D_CODE11

The structure that saves setup information of the Symbology supported by 1D Barcode Scanner.

```
Public structCONFIG_1D_CODE11
{
  boolbEnabled;
  bool bCheckDigitVerify;
  bool bXmitCheckDigit;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**

whether Code 11 is enabled or disabled

- **bCheckDigitVerify**

Sets the number of check digits at the end of the bar code that verify the integrity of the data. At least one checkdigit is always required. Check digits are not automatically transmitted with the data.

- **bXmitCheckDigit**

whetherCode 11 Check Digit is transmitted with the data.

- **nMaxLength**
  max length for Code 11
- **nMinLength**
  min length for Code 11

1.2.16 CONFIG_2DSWD_AUSPOST

The structure that saves setup information of the Australian Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_AUSPOST
{
  boolbEnabled;
}
```

- **bEnabled**
  whether Australian Postal Code is enabled or disabled

1.2.17 CONFIG_2DSWD_AZTEC

The structure that saves setup information of the Aztec Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_AUSPOST
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether AZTEC is enabled or disabled
- **nMaxLength**
  max length for AZTEC
- **nMinLength**
  min length for AZTEC

1.2.18 CONFIG_2DSWD_BPO

The structure that saves setup information of the British Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_BPO
{
  boolbEnabled;
}
```

- **bEnabled**
  whether British Post is enabled or disabled

1.2.19 CONFIG_2DSWD_CANPOST

The structure that saves setup information of the Canadian Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CANPOST
{
  boolbEnabled;
}
```

- **bEnabled**
  whether Canadian Post is enabled or disabled

1.2.20 CONFIG_2DSWD_CHINAPOST

The structure that saves setup information of the Chinese Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CHINAPOST
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether China post is enabled or disabled
- **nMaxLength**
  max length for China post
- **nMinLength**
  min length for China post

1.2.21 CONFIG_2DSWD_CODABAR

The structure that saves setup information of the CodabarSymbology

Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODABAR
{
  boolbEnabled;
  bool bCheckCharOn;
  bool bSSXmit;
  bool bXmitCheckChar;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
- **whether Codabar is enabled or disabled**
- **bCheckCharOn**

  BOOL variable that determines if the engine will read Codabar bar codes with or without check characters.If TRUE, the engine only decodes Codabar codes with a check character. If FALSE, the decoder decodes codes withor without a check character.
- **bSSXmit**

  BOOL variable that determines if the start and stop characters are returned in the data string after asuccessful Codabar decode. If bSSXmit is TRUE, the start and stop characters are included. If FALSE, they are notincluded.
- **bXmitCheckChar**

  BOOL variable that determines if the engine will return the check character as part of the data string after asuccessful decode. If TRUE, the engine returns the check character. If FALSE the check character is not returned.
- **nMaxLength**

  max length for Codabar
- **nMinLength**

  min length for Codabar

1.2.22 CONFIG_2DSWD_CODABLOCK

The structure that saves setup information of the Codablock F Symbology
Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODABLOCK
{
  boolbEnabled;
```

```
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whetherCodablock F is enabled or disabled
- **nMaxLength**
  max length for Codablock F
- **nMinLength**
  min length for Codablock F

1.2.23 CONFIG_2DSWD_CODE11

The structure that saves setup information of the Code 11 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE11
{
  boolbEnabled;
  bool bTwoCheckDigits;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whetherCodablock F is enabled or disabled
- **bTwoCheckDigits**
  BOOL variable that determines if the engine is decoding Code 11 bar codes that have two check digits. IfTRUE, the engine is decoding Code 11 bar codes that have two check digits. If FALSE the engine decodes Code 11bar codes as if they were printed with only one check digit.
- **nMaxLength**
  max length for Code 11
- **nMinLength**
  min length for Code 11

1.2.24 CONFIG_2DSWD_CODE128

The structure that saves setup information of the Code 128 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE128
{
```

```
    boolbEnabled;
    short nMaxLength;
    short nMinLength;
}
```

- **bEnabled**
  whether Code 128 is enabled or disabled
- **nMaxLength**
  max length for Code 128
- **nMinLength**
  min length for Code 128

1.2.25 CONFIG_2DSWD_CODE16K

The structure that saves setup information of the Code 16K Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE16K
{
    boolbEnabled;
    short nMaxLength;
    short nMinLength;
}
```

- **bEnabled**
  whether Code 16K is enabled or disabled
- **nMaxLength**
  max length for Code 16K
- **nMinLength**
  min length for Code 16K

1.2.26 CONFIG_2DSWD_CODE32

The structure that saves setup information of the Code 32 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE32
{
    boolbEnabled;
}
```

- **bEnabled**
  whether Code 16K is enabled or disabled

1.2.27 CONFIG_2DSWD_CODE39

The structure that saves setup information of the Code 39 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE39
{
  boolbEnabled;
  bool bAppend;
  bool bCheckCharOn;
  bool bFullAscii;
  bool bSSXmit;
  bool bXmitCheckChar;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Code 39 is enabled or disabled
- **bAppend**
  BOOL variable that determines if the engine should append together and buffer up Code 39 symbols thatstart with a space (excluding the start and stop characters). The engine stores the symbols in the order in which theyare read. It returns the data after a Code 39 symbol with no leading space is read. The return data has the leadingspaces removed. If TRUE, the append feature is enabled. If FALSE, the append feature is disabled.
- **bCheckCharOn**
  BOOL variable that determines if the engine will read Code 39 bar codes with or without check characters.If TRUE, the engine only decodes Code 39 codes with a check character. If FALSE, the decoder decodes codes withor without a check character.
- **bFullAscii**
  BOOL variable that determines if certain character pairs within the bar code symbol are interpreted andreturned as a single character. If bFullAscii is TRUE, interpretation is enabled. If FALSE, no interpretation is attempted.
- **bSSXmit**
  BOOL variable that determines if the start and stop characters are returned in the data string after asuccessful Code 39 decode. If bSSXmit is TRUE, the start and stop characters are included. If FALSE, they are notincluded.

- **bXmitCheckChar**
  BOOL variable that determines if the engine will return the check character as part of the data string after asuccessful decode. If TRUE, the engine returns the check character. If FALSE, the check character is not returned.
- **nMaxLength**
  max length for Code 39
- **nMinLength**
  min length for Code 39

1.2.28 CONFIG_2DSWD_CODE49

The structure that saves setup information of the Code 49 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE49
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Code 49 is enabled or disabled
- **nMaxLength**
  max length for Code 49
- **nMinLength**
  min length for Code 49

1.2.29 CONFIG_2DSWD_CODE93

The structure that saves setup information of the Code 93 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_CODE93
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Code 93 is enabled or disabled

- **nMaxLength**
  max length for Code 93
- **nMinLength**
  min length for Code 93

1.2.30 CONFIG_2DSWD_COMPOSITE

The structure that saves setup information of the Composite Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_COMPOSITE
{
  boolbEnabled;
  bool bCompositeOnUpcEan;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Composite is enabled or disabled
- **bCompositeOnUpcEan**
  BOOL variable that contains the enabled state of EAN•UCC Composite code associated with EAN and UPCcodes.
- **nMaxLength**
  max length for Composite
- **nMinLength**
  min length for Composite

1.2.31 CONFIG_2DSWD_COUPONCODE

The structure that saves setup information of the Coupon Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_COUPONCODE
{
  boolbEnabled;
}
```

- **bEnabled**
  whether Coupon code is enabled or disabled

1.2.32 CONFIG_2DSWD_DATAMATRIX

The structure that saves setup information of the Data Matrix Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_DATAMATRIX
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Data Matrix is enabled or disabled
- **nMaxLength**
  max length for Data Matrix
- **nMinLength**
  min length for Data Matrix

1.2.33 CONFIG_2DSWD_DUTCHPOST

The structure that saves setup information of the KIX(Netherlands) Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_DUTCHPOST
{
  boolbEnabled;
}
```

- **bEnabled**
  whether KIX(Netherlands) Post is enabled or disabled

1.2.34 CONFIG_2DSWD_EAN13

The structure that saves setup information of the Ean-13 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_EAN13
{
  boolbEnabled;
  bool bAddenda2Digit;
  bool bAddenda5Digit;
  bool bAddendaReq;
```

```
    bool bAddendaSeparator;
    bool bXmitCheckChar;
}
```

- **bEnabled**
  whether EAN 13 is enabled or disabled
- **bAddenda2Digit**
  BOOL variable that determines if the engine will look for a 2 digit addenda at the end of the EAN/JAN-13 barcode. If TRUE, and an addenda is present, the engine adds the two digit addenda data to the end of the message. IfFALSE, the engine ignores addenda data.
- **bAddenda5Digit**
  BOOL variable that determines if the engine will look for a 5 digit addenda at the end of the EAN/JAN-13 barcode. If TRUE, and an addenda is present, the engine adds the five digit addenda data to the end of the message. IfFALSE, the engine ignores addenda data.
- **bAddendaReq**
  BOOL variable that determines if the engine will decode only EAN/JAN-13 bar codes that have a 2 or 5 digitaddenda.If TRUE, the engine decodes only EAN symbols with an addenda. If FALSE, the engine decodes all enabledEAN/JAN-13 symbols.
- **bAddendaSeparator**
  BOOL variable that determines if there is a space character between the data from the bar code and thedata from the addenda.If TRUE, there is a space. If FALSE, there is no space.
- **bXmitCheckChar**
  BOOL variable that determines if the engine will return the check character as part of the data string after asuccessful decode. If TRUE, the engine returns the check character. If FALSE, the check character is not returned.

1.2.35 CONFIG_2DSWD_ENA8

The structure that saves setup information of the Ean-8 Symbology
Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_EAN8
{
    boolbEnabled;
    bool bAddenda2Digit;
    bool bAddenda5Digit;
    bool bAddendaReq;
```

```
    bool bAddendaSeparator;
    bool bXmitCheckChar;
}
```

- **bEnabled**
  whether EAN 13 is enabled or disabled
- **bAddenda2Digit**
  BOOL variable that determines if the engine will look for a 2 digit addenda at the end of the EAN/JAN-8 barcode. If TRUE, and an addenda is present, the engine adds the two digit addenda data to the end of the message. IfFALSE, the engine ignores addenda data.
- **bAddenda5Digit**
  BOOL variable that determines if the engine will look for a 5 digit addenda at the end of the EAN/JAN-8 barcode. If TRUE, and an addenda is present, the engine adds the five digit addenda data to the end of the message. IfFALSE, the engine ignores addenda data.
- **bAddendaReq**
  BOOL variable that determines if the engine will decode only EAN/JAN-8 bar codes that have a 2 or 5 digitaddenda.If TRUE, the engine decodes only EAN/JAN-8 symbols with an addenda. If FALSE, the engine decodes allenabled EAN/JAN-8 symbols.
- **bAddendaSeparator**
  BOOL variable that determines if there is a space character between the data from the bar code and thedata from the addenda.If TRUE, there is a space. If FALSE, there is no space.
- **bXmitCheckChar**
  BOOL variable that determines if the engine will return the check character as part of the data string after asuccessful decode. If TRUE, the engine returns the check character. If FALSE, the check character is not returned.

1.2.36 CONFIG_2DSWD_GENERICCODE128

The structure that saves setup information of the Generic Code 128 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_GENERICCODE128
{
    boolbEnabled;
    short nMaxLength;
    short nMinLength;
```

}

- **bEnabled**
  whether Generic Code 128 is enabled or disabled
- **nMaxLength**
  max length for Generic Code 128
- **nMinLength**
  min length for Generic Code 128

1.2.37 CONFIG_2DSWD_GS1128

The structure that saves setup information of the GS1 128 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_GS1128
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether GS1 128 is enabled or disabled
- **nMaxLength**
  max length for GS1 128
- **nMinLength**
  min length for GS1 128

1.2.38 CONFIG_2DSWD_IATA25

The structure that saves setup information of the IATA 2 of 5 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_IATA25
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Straight 2 of 5 IATA is enabled or disabled
- **nMaxLength**

max length for Straight 2 of 5 IATA

- **nMinLength**

min length for Straight 2 of 5 IATA

1.2.39 CONFIG_2DSWD_INT25

The structure that saves setup information of the Interleaved 2 of 5 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_INT25
{
  boolbEnabled;
  bool bCheckDigitOn;
  bool bXmitCheckDigit;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**

whether Straight 2 of 5 IATA is enabled or disabled

- **bCheckDigitOn**

BOOL variable that determines if the engine will read Interleaved 2 of 5 bar codes with or without checkcharacters.If TRUE, the engine only decodes Interleaved 2 of 5 codes with a check digit. If FALSE, the decoderdecodes codes with or without a check digit.

- **bXmitCheckDigit**

BOOL variable that determines if the engine will return the check digit as part of the data string after asuccessful decode. If TRUE, the engine returns the check digit. If FALSE, the check digit is not returned.

- **nMaxLength**

max length for Straight 2 of 5 IATA

- **nMinLength**

min length for Straight 2 of 5 IATA

1.2.40 CONFIG_2DSWD_ISBT

The structure that saves setup information of the ISBT Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_ISBT
{
```

```
    boolbEnabled;
}
```

- **bEnabled**
  whether ISBT 128 is enabled or disabled

1.2.41 CONFIG_2DSWD_JAPOST

The structure that saves setup information of the Japanese Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_JAPOST
{
    boolbEnabled;
}
```

- **bEnabled**
  whether Japanese Post is enabled or disabled

1.2.42 CONFIG_2DSWD_KOREAPOST

The structure that saves setup information of the Korean Post Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_KOREAPOST
{
    boolbEnabled;
    short nMaxLength;
    short nMinLength;
}
```

- **bEnabled**
  whether Korean Post is enabled or disabled
- **nMaxLength**
  max length for Korean Post
- **nMinLength**
  min length for Korean Post

1.2.43 CONFIG_2DSWD_MAXICODE

The structure that saves setup information of the Maxi Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MAXICODE
{
  boolbEnabled;
  bool bCarrierMsgOnly;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Maxi Code is enabled or disabled
- **bCarrierMsgOnly**
  This parameter is ignored and no longer supported.
- **nMaxLength**
  max length for Maxi Code
- **nMinLength**
  min length for Maxi Code

1.2.44 CONFIG_2DSWD_MAXICODE

The structure that saves setup information of the Maxi Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MAXICODE
{
  boolbEnabled;
  bool bCarrierMsgOnly;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Maxi Code is enabled or disabled
- **bCarrierMsgOnly**
  This parameter is ignored and no longer supported.
- **nMaxLength**
  max length for Maxi Code
- **nMinLength**
  min length for Maxi Code

1.2.45 CONFIG_2DSWD_MESA

The structure that saves setup information of the Mesa Symbology

Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MESA
{
  boolb1MSEnabled;
  bool b3MSEnabled;
  bool b9MSEnabled;
  bool bEMSEnabled;
  bool bIMSEnabled;
  bool bUMSEnabled;
}
```

- **b1MSEnabled**
  BOOL variable that contains the enabled state of Code 128 Mesa.
  TRUE = Enabled, FALSE = Disabled.
- **b3MSEnabled**
  BOOL variable that contains the enabled state of Code 39 Mesa.
  TRUE = Enabled, FALSE = Disabled.
- **b9MSEnabled**
  BOOL variable that contains the enabled state of Code 93 Mesa.
- **bEMSEnabled**
  BOOL variable that contains the enabled state of EAN13 Mesa.
  TRUE = Enabled, FALSE = Disabled.
- **bIMSEnabled**
  BOOL variable that contains the enabled state of Interleaved 2 of 5 Mesa.
  TRUE = Enabled, FALSE =Disabled.
- **bUMSEnabled**
  BOOL variable that contains the enabled state of UPCA Mesa.
  TRUE = Enabled, FALSE = Disabled.

1.2.46 CONFIG_2DSWD_MICROPDF

The structure that saves setup information of the Micro PDF
Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MICROPDF
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
```

}

- **bEnabled**
  whether Micro PDF is enabled or disabled
- **nMaxLength**
  max length for Micro PDF
- **nMinLength**
  min length for Micro PDF

1.2.47 CONFIG_2DSWD_MSI

The structure that saves setup information of the MSI Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MSI
{
  boolbEnabled;
  bool bXmitCheckChar;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Maxi Code is enabled or disabled
- **bXmitCheckChar**
  BOOL variable that determines if the engine will return the check character as part of the data string after asuccessful decode. If TRUE, the engine returns the check character. If FALSE, the check character is not returned.
- **nMaxLength**
  max length for MSI
- **nMinLength**
  min length for MSI

1.2.48 CONFIG_2DSWD_MX25

The structure that saves setup information of the Micro PDF Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_MX25
{
  boolbEnabled;
  short nMaxLength;
```

```
    short nMinLength;
}
```

- **bEnabled**
  whether Matrix 2 of 5 is enabled or disabled
- **nMaxLength**
  max length for Matrix 2 of 5
- **nMinLength**
  min length for Matrix 2 of 5

1.2.49 CONFIG_2DSWD_OCR

The structure that saves setup information of the OCR Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_OCR
{
    String CheckChar;
    String GroupG;
    String GroupH;
    String Template;
    OCRMode nFont;
}
```

- **CheckChar**
  A null-terminated string that represents a check character position in the template strings.
- **GroupG**
  A null-terminated string that represents a list of characters that can be substituted for the lower-case 'g' in the templatestrings.
- **GroupH**
  A null-terminated string that represents a list of characters that can be substituted for the lower-case 'h' in the templatestrings.
- **Template**
  A null-terminated string that indicates one or more template patterns for the OCR decode. The following characters areallowed:
  A-Z :capital letters are matched as is
  d : a digit from 0 - 9
  a : alphanumeric character
  l : alphabetic letter
  g : any character specified in group G
  h : any character specified in group H

1.2.50 CONFIG_2DSWD_PDF417

The structure that saves setup information of the PDF417 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_PDF417
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether PDF417 is enabled or disabled
- **nMaxLength**
  max length for PDF417
- **nMinLength**
  min length for PDF417

1.2.51 CONFIG_2DSWD_PLANET

The structure that saves setup information of the Planet Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_PLANET
{
  boolbEnabled;
  boolbXmitCheckDigit;
}
```

- **bEnabled**
  whether Planet is enabled or disabled
- **bXmitCheckDigit**
  BOOL variable that determines if the engine will return the check digit as part of the data string after asuccessful decode. If TRUE, the engine returns the check digit. If FALSE, the check digit is not returned.

1.2.52 CONFIG_2DSWD_PLESSEY

The structure that saves setup information of the Plessey Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_PLESSEY
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether Plessey Code is enabled or disabled
- **nMaxLength**
  max length for Plessey Code
- **nMinLength**
  min length for Plessey Code

1.2.53 CONFIG_2DSWD_POSICODE

The structure that saves setup information of the Posi Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_POSICODE
{
  boolbEnabled;
  short wLimited;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whetherPosi Code is enabled or disabled
- **wLimited**
  short variable that reflects if Posicode Limited A or Posicode Limited B decoding is enabled. A value of1 indicates Posicode Limited A is enabled, and a value of 2 indicates Posicode Limited B decoding is enabled. A valueof 0 indicates that decoding of both Limited A and Limited B is disabled.
- **nMaxLength**
  max length for Posi Code
- **nMinLength**
  min length for Posi Code

1.2.54 CONFIG_2DSWD_POSTNET

The structure that saves setup information of the PostnetSymbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_POSTNET
{
  boolbEnabled;
  boolbXmitCheckDigit;
}
```

- **bEnabled**
  whetherPostnet is enabled or disabled
- **bXmitCheckDigit**
  BOOL variable that determines if the engine will return the check digit as part of the data string after asuccessful decode. If TRUE, the engine returns the check digit. If FALSE, the check digit is not returned.

1.2.55 CONFIG_2DSWD_QR

The structure that saves setup information of the QR Code Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_QR
{
  boolbEnabled;
  short nMaxLength;
  short nMinLength;
}
```

- **bEnabled**
  whether QR Code is enabled or disabled
- **nMaxLength**
  max length for QR Code
- **nMinLength**
  min length for QR Code

1.2.56 CONFIG_2DSWD_RSS

The structure that saves setup information of the RSS Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_RSS
{
  boolbEnabled;
  short nMaxLength;
```

```
    short nMinLength;
}
```

- **bEnabled**
  whether RSS is enabled or disabled
- **nMaxLength**
  max length for RSS
- **nMinLength**
  min length for RSS

1.2.57 CONFIG_2DSWD_STRT25

The structure that saves setup information of the Straight 2 of 5 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_STRT25
{
    boolbEnabled;
    short nMaxLength;
    short nMinLength;
}
```

- **bEnabled**
  whether Straight 2 of 5 is enabled or disabled
- **nMaxLength**
  max length for Straight 2 of 5
- **nMinLength**
  min length for Straight 2 of 5

1.2.58 CONFIG_2DSWD_TELEPEN

The structure that saves setup information of the TelepenSymbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_STRT25
{
    boolbEnabled;
    bool bOldStyle;
    short nMaxLength;
    short nMinLength;
}
```

- **bEnabled**

whetherTelepen is enabled or disabled

- **bOldStyle**

BOOL variable that reflects if the engine is configured to reads Telepen labels that were encoded with eitherthe original or the AIM specification.

- **nMaxLength**

max length for Telepen

- **nMinLength**

min length for Telepen

1.2.59 CONFIG_2DSWD_TLC39

The structure that saves setup information of the TLC39 Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_TLC39
{
  boolbEnabled;
}
```

- **bEnabled**

whether TLC39 is enabled or disabled

1.2.60 CONFIG_2DSWD_TRIOPTIC

The structure that saves setup information of the TriopticSymbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_TRIOPTIC
{
  boolbEnabled;
}
```

- **bEnabled**

whetherTrioptic is enabled or disabled

1.2.61 CONFIG_2DSWD_UPCA

The structure that saves setup information of the UPC-A Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_UPCA
{
```

```
    boolbEnabled;
    bool bAddenda2Digit;
    bool bAddenda5Digit;
    bool bAddendaReq;
    bool bAddendaSeparator;
    bool bXmitCheckDigit;
    bool bXmitNumSys;
}
```

- **bEnabled**

  whether UPC-A is enabled or disabled

- **bAddenda2Digit**

  BOOL variable that determines if the engine will look for a 2 digit addenda at the end of the UPC bar code.If TRUE, and an addenda is present, the engine adds the two digit addenda data to the end of the message. If FALSE,the engine ignores addenda data.the original or the AIM specification.

- **bAddenda5Digit**

  BOOL variable that determines if the engine will look for a 5 digit addenda at the end of the UPC bar code.If TRUE, and an addenda is present, the engine adds the five digit addenda data to the end of the message. If FALSE,the engine ignores addenda data.

- **bAddendaReq**

  BOOL variable that determines if the engine will decode only UPC bar codes that have a 2 or 5 digit addenda.If TRUE, the engine decodes only UPC symbols with an addenda. If FALSE, the engine decodes all enabled UPCsymbols.

- **bAddendaSeparator**

  BOOL variable that determines if there is a space character between the data from the bar code and thedata from the addenda.If TRUE, there is a space. If FALSE, there is no space.

- **bXmitCheckDigit**

  BOOL variable that determines if the engine will return the check digit as part of the data string after asuccessful decode. If TRUE, the engine returns the check digit. If FALSE, the check digit is not returned.

- **bXmitNumSys**

  BOOL variable that determines if the engine will return the numeric system digit of the UPC label.If TRUE,the engine returns the number system digit. If FALSE, the number system digit is not returned.

1.2.62 CONFIG_2DSWD_UPCE

The structure that saves setup information of the UPC-E Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_UPCE
{
  bool bAddenda2Digit;
  bool bAddenda5Digit;
  bool bAddendaReq;
  bool bAddendaSeparator;
  bool bE0Enabled;
  bool bE1Enabled;
  bool bExpandVersionE;
  bool bXmitCheckDigit;
  bool bXmitNumSys;
}
```

- **bAddenda2Digit**
  BOOL variable that determines if the engine will look for a 2 digit addenda at the end of the UPC bar code.If TRUE, and an addenda is present, the engine adds the two digit addenda data to the end of the message. If FALSE,the engine ignores addenda data.
- **bAddenda5Digit**
  BOOL variable that determines if the engine will look for a 5 digit addenda at the end of the UPC bar code.If TRUE, and an addenda is present, the engine adds the five digit addenda data to the end of the message. If FALSE,the engine ignores addenda data.
- **bAddendaReq**
  BOOL variable that determines if the engine will decode only UPC bar codes that have a 2 or 5 digit addenda.If TRUE, the engine decodes only UPC symbols with an addenda. If FALSE, the engine decodes all enabled UPCsymbols.
- **bAddendaSeparator**
  BOOL variable that determines if there is a space character between the data from the bar code and thedata from the addenda.If TRUE, there is a space. If FALSE, there is no space.
- **bXmitCheckDigit**
  BOOL variable that determines if the engine will return the check digit as part of the data string after asuccessful decode. If TRUE, the engine returns the check digit. If FALSE, the check digit is not returned.
- **bXmitNumSys**

BOOL variable that determines if the engine will return the numeric system digit of the UPC label.If TRUE,the engine returns the number system digit. If FALSE, the number system digit is not returned.

1.2.63 CONFIG_2DSWD_USPS4CB

The structure that saves setup information of the 4-State Customer Barcode Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_USPS4CB
{
  boolbEnabled;
}
```

- **bEnabled**

  whether USPS 4-State Customer Barcode is enabled or disabled

1.2.64 CONFIG_2DSWD_IDTAG

The structure that saves setup information of the IDTAG Symbology Supported by 2D Barcode Scanner.

```
Public structCONFIG_2DSWD_IDTAG
{
  boolbEnabled;
}
```

- **bEnabled**
  whether IDTAG is enabled or disabled

1.3 Delegates

1.3.1 BARCODECALLBACK

Calling delegate function, while successfully read barcode.
In order to process the barcode data in application program, need to use SetCallback (BARCODECALLBACK pFunc) function to register delegate function.

```
Public delegate void BARCODECALLBACK();
```

1.4 Methods

1.4.1 Open

Supply power to the Barcode device, and initialize the Barcode device. To be assigned necessary system resource.

```
BARCODE_RESULT  Open();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

1.4.2 Close

Remove voltage from Barcode device and clear the assigned resource.

```
BARCODE_RESULT  Close();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

1.4.3 IsOpened

Checking whether the Barcode device is already opened.If BarcodeOpen() was successfully executed, the Barcode will be under open status.

```
BOOLIsOpened()
```

Parameters

None

Return Values

If the Barcode under the status of open, TRUE will be returned,

If under the status of close, FALSE will be returned.

1.4.4 SetCallback

When scanning has been done completely, register the application program's delegate function that called by Barcode system driver.

```
BARCODE_RESULTSetCallback (
    BARCODECALLBACK pFunc,
);
```

Parameters

pFunc

BARCODECALLBACK's object

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

1.4.5 GetVersionInfo

Reading the version of Barcode Scanner.

Not currently used.

```
BARCODE_RESULT    GetVersionInfo ( );
```

Parameters

None

Return Values

This function always return the BARCODE_RESULT_UNSUPPORTED.

1.4.6 Start

Starting to scan Barcode.

```
BARCODE_RESULT    Start();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

If successfully excuted, reading Barcode will succeed or trying reading Barcode before Stop function is called. If reading Barcode successfully finishs, execute the BARCODECALLBACK delegate.

1.4.7 StartScanRawData

Starting to read Barcode asynchronously. The resulf of reading is not string but the arrangement of byte.

```
BARCODE_RESULT    StartScanRawData();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

If successfully excuted, reading Barcode will succeed or trying reading Barcode before Stop function is called. If reading Barcode successfully finishs, execute the BARCODECALLBACK delegate.

Only in support of 2D Barcode scanner.

1.4.8 Stop

Stop to scan Barcode.

```
BARCODE_RESULT    Stop();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

1.4.9 StopScanRawData

Stop reading Barcode.

```
BARCODE_RESULT    StopScanRawData();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

1.4.10 GetScannedData

Reading barcode information which recognized successfully from Barcode scanner device.

```
BARCODE_RESULTGetScannedData (
    ref String BarcodeValue,
    ref String BarcodeTypeName,
    ref String BarcodeTypeId,
);
```

Parameters

BarcodeValue

string parameter which will be stored barcode value.

BarcodeTypeName

string parameter which will be stored barcode Type Name.

BarcodeTypeId

string parameter that will store barcode Type Id.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

This function must be called from within registered delegate function by user.

1.4.11 GetBarcodeRawData

Getting raw data from recognized Barcode.

```
BARCODE_RESULT    GetBarcodeRawData (
    ref byte[] BarcodeValue,
    ref int Length,
    ref String BarcodeTypeName,
    ref String BarcodeTypeId,
```

Parameters

BarcodeValue

The arrangement of byte in which raw data of Barcode will be stored

Length

The length of stored raw data

BarcodeTypeName

Barcode Type Name

BarcodeTypeId

Barcode Type ID

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

This function must be called from within BARCODECALLBACK delegate function by user.

1.4.12 InitCapture

In order to capture images through 2D Barcode, assigned necessary system resource and initialize related parameters.

BARCODE_RESULTInitCapture ();

Parameters

None

Return Values

BARCODE_RESULT_SUCCESS will be returned if the capture initializing executes properly. BARCODE_RESULT_FAILURE will be returned if failed.

BARCODE_RESULT_UNSUPPORTED will be returned if call this function in a 1D BARCODE PDA.

Notes

Only in support of 2D Barcode scanner

1.4.13 DeinitCapture

In order to capture image, clear the assigned system resource.

BARCODE_RESULTDeinitCapture ();

Parameters

None

Return Values

BARCODE_RESULT_SUCCESS will be returned if the system resource cleared properly.
BARCODE_RESULT_FAILURE will be returned if failed.
BARCODE_RESULT_UNSUPPORTED will be returned if call this function in a 1D Barcode PDA.

Notes

Only in support of 2D Barcode scanner.

1.4.14 SetPreviewHwnd

Set a preview window's handle.

```
BARCODE_RESULTSetPreviewHwnd (
    IntPtrpHandle,
);
```

Parameters

pHandle

Handle to preview window.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of 2D Barcode scanner.

1.4.15 StartPreview

Drawing the image data that transmitted from Barcode to preview window screen.

```
BARCODE_RESULTStartPreview ();
```

Parameters

None

Return Values

BARCODE_RESULT_SUCCESS will be returned if the preview window starting properly.
BARCODE_RESULT_FAILURE will be returned if failed.
BARCODE_RESULT_UNSUPPORTED will be returned if call this function in a 1D Barcode PDA.

Notes

Only in support of 2D Barcode scanner.

1.4.16 StopPreview

Stop to preview window.

```
BARCODE_RESULTStopPreview ();
```

Parameters

None

Return Values

BARCODE_RESULT_SUCCESS will be returned if stop to preview window execute properly.

BARCODE_RESULT_UNSUPPORTED will be returned if call this function in a 1D BARCODE PAD.

Notes

Only in support of 2D Barcode scanner.

1.4.17 DoCapture

Capture image data and store into files that through Barcode.

```
BARCODE_RESULTDoCapture (
    refBARCODECAPTUREPARAMSCaputreParams
);
```

Parameters

CaptureParams

transmit image's resolution, format, and storage path.

Return Values

BARCODE_RESULT_SUCCESS will be returned if captured image successfully.

BARCODE_RESULT_UNSUPPORTED will be returned if call this function in a 1D BARCODE PDA.

Notes

Only in support of 2D Barcode scanner.

1.4.18 GetEnableStateAll

Returning Enable/Disable condition about all of symbology.

```
BARCODE_RESULT   GetEnableStateAll (
    refbool[] bSymbolTable
);
```

Parameters

bSymbolTable

The arrangement of bool in which Enable/Disable condition will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES, in case of OCR, due to the property of Symbology, it is so difficult to check condition that directly reading setting value(GetSymbologyConfig) is recommended.

1.4.19 EnableSymbologiesAll

Setting Enable/Disable condition about all of symbology.

```
BARCODE_RESULT   GetEnableStateAll (
      bool bEnable
);
```

Parameters

bEnable

The Enable/Disable condition that will be applied to all symbology.

True: all symbology enabled

False:all symbology disabled

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.4.20 EnableSymbologies

Enabling the specific Symbology.

```
BARCODE_RESULTEnableSymbologies (
      bool[] bSymbolTable
);
```

Parameters

bSymbolTable

the arrangement of bool in which information of symbology to be enabled is stored.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES. Set the symbology to be enabled to "true" using value defined in SYMBOLOGIES_1D or SYMBOLOGIES_2DSWD.

1.4.21 DisableSymbologies

Disabling the specific Symbology

```
BARCODE_RESULT   DisableSymbologies (
        bool[] bSymbolTable
);
```

Parameters

bSymbolTable

the arrangement of bool in which information of symbology to be disabled is stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES. Set the symbology to be enabled to "true" using value defined in SYMBOLOGIES_1D or SYMBOLOGIES_2DSWD.

1.4.22 GetSymbologyConfig

Getting the set value of specific Symbology from Scanner.

```
BARCODE_RESULT   GetSymbologyConfig (
        refSYMBOL_CONFIG SymbolConfig
);
```

Parameters

SymbolConfig

The structure where the set value of Symbology imported from Scanner will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Reference to 1.2.3 SYMBOL_CONFIG

1.4.23 SetSymbologyConfig

Storing the set value of specific Symbology in Scanner.

```
BARCODE_RESULT   SetSymbologyConfig (
      SYMBOL_CONFIG SymbolConfig
);
```

Parameters

SymbolConfig

The structure where the set value of Symbology to be stored in Scanner is stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.2.3 SYMBOL_CONFIG 참조

Reference to 1.2.3 SYMBOL_CONFIG

1.4.24 GetGeneralConfig

Getting set value of 1D Barcode Scanner like security level, scanning angle.

```
BARCODE_RESULT   GetGeneralConfig (
      refGENERAL_CONFIGGeneralConfig
);
```

Parameters

GeneralConfig

The structure in which set value imported from Scanner will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of 1D Barcode scanner.

Reference to 1.2.2 GENERAL_CONFIG

1.4.25 SetGeneralConfig

Setting value of 1D Barcode Scanner like security level, scanning angle

```
BARCODE_RESULT   SetGeneralConfig (
      GENERAL_CONFIGGeneralConfig
);
```

Parameters

GeneralConfig

The structure in which set value that will be stored in Scanner is Stored.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of 1D Barcode scanner.

Reference to 1.2.2 GENERAL_CONFIG

1.4.26 SetDefaultSymbol

Resetting all set values of Symbology

```
BARCODE_RESULT    SetDefaultSymbol ( );
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.4.27 SetScanningLightMode

Setting LED light(aimers, illumination) mode while 2D Barcode Scanner scanning.

```
BARCODE_RESULT    SetScanningLightMode (
        int nMode
);
```

Parameters

nMode

0 : Neither aimers nor illumination

1 : Illumination only

2 : Aimers only

3 : Both aimers and illumination

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.4.28 SetLeaveLightsOn

Setting on/off switch value of LED light(aimers, illumination) while 2D Barcode Scanner

scanning.

```
BARCODE_RESULT   SetLeaveLightsOn (
        boolbEnable
);
```

Parameters

bEnable

true: LED light on while scanning.

False: LED light on and off while scanning

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.4.29 GetLeaveLightsOn

Getting on/off value of LED light(aimers, illumination) while 2D Barcode Scanner scanning.

```
BARCODE_RESULT   GetLeaveLightsOn (
        ref boolbEnable
);
```

Parameters

bEnable

true: LED light on while scanning.

False: LED light on and off while scanning

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

1.5 Properties

1.5.1 IlluminationPowerLevel

Get or set the Illumination Power value of 2D Scanner Engine.

```
byteIlluminationPowerLevel{ get; set;}
```

Value

Power level (min:0 ~ max:255)

2 C/C++ API Reference

2.1 Enumerations

2.1.1 BARCODE_RESULT

The result of a call to functions.

- **BARCODE_RESULT_ALREADY_BARCODE_SELECTED**
  Barcode device was already selected.
- **BARCODE_RESULT_ALREADY_OPENED**
  Barcode device was already opened.
- **BARCODE_RESULT_BARCODE_INIT_FAILURE**
  Barcode device initialization failed.
- **BARCODE_RESULT_CAM_SELECTED**
  CAM device was already selected.
- **BARCODE_RESULT_FAILURE**
  Function execution failed.
- **BARCODE_RESULT_INVALID_ARGS**
  Invalid paramerter.
- **BARCODE_RESULT_INVALID_DEVICE**
  There is no barcode device.
- **BARCODE_RESULT_NOT_OPENED**
  Calling function without open.
- **BARCODE_RESULT_OUTOFMEMORY**
  Failed to assign memory.
- **BARCODE_RESULT_SUCCESS**
  Function execution success.
- **BARCODE_RESULT_UNSUPPORTED**
  Unsupported command.

2.2 Constants

2.2.1 WM_BARCODE_ONSCANED

Windows Message, that will process while get data from module.

- **#define WM_BARCODE_ONSCANED WM_USER + 1801**

2.2.2 CAP_IMG_VGA

640 * 480

- **#define CAP_IMG_VGA 1**

2.2.3 CAP_IMG_QVGA

320 * 240

- **#define CAP_IMG_VGA 2**

2.2.4 CAP_IMG_PREVIEW

240 * 180

- **#define CAP_IMG_VGA 3**

2.2.5 IMG_FORMAT_JPEG

Jpeg format

- **#define IMG_FORMAT_JPEG 1**

2.2.6 IMG_FORMAT_BMP

BMP format

- **#define IMG_FORMAT_BMP 2**

2.3 Structures

2.3.1 BARCODECAPTUREPARAMS

The structure that while capturing images through 2D Barcode scanner.

```
typedefstruct
{
  charImgFormat
  charImgSize;
  TCHARszFilePath[MAX_PATH];
}BARCODECAPTUREPARAMS;
```

- **ImgFormat**
  format.
- **ImgSize**
  size.
- **szFilePath**
  the path of image storage(include file name).

2.3.2 BARCODE_GENERAL_CONFIG

The structure that is used when reading or setting the condition of security level,

scanning angle of 1D Barcode Scanner.

```
typedef struct
{
  BOOL bBiDirectionalRedundancy;
  int nAngle;
  int nLinearCodeSecurityLevel
}BARCODE_GENERAL_CONFIG;
```

- **bBiDirectionalRedundancy**
  Returns whether a bar code must be successfully scanned in both direction(forward and reverse) before being decoded.
- **nAngle**
  Scanning angle. 5=Narrow, 6=Wide
- **nLinearCodeSecurityLevel**
  Four levels(1~4) of decode security for linear code types are supported. Select a highger security level for low levels of bar code quality. As security levels increase, the decoder's aggressivenese decrease.

2.3.3 BARCODE_SYMBOL_CONFIG

The structure in which setting information is stored for Symbology setting

```
typedef struct
{
  Void * pConfigData;
  Int Symbol;
}BARCODE_SYMBOL_CONFIG;
```

- **pConfigData**
  The pointer of structure of Symbology that will change setting
- **Symbol**
  ID of Symbology that change setting

2.3.4 Symbol Configuration Structures

Omitted because of the same content with NET API reference
Reference to the lists of Net API reference from 1.2.4 to 1.2.64.

2.4 Callback function definition

2.4.1 BARCODECALLBACK

Callback function, that will process while get data from module

In order to process the barcode data in application program, need to use BarcodeSetCallback (BARCODECALLBACK pFunc) function to register delegate function.

- **typedef void (CALLBACK* BARCODECALLBACK)();**

2.5 Methods

2.5.1 BarcodeOpen

Supply power to the Barcode device, and initialize the Barcode device. To be assigned necessary system resource.

BARCODE_RESULTBarcodeOpen();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

2.5.2 BarcodeClose

Remove voltage from Barcode device and clear the assigned resource.

BARCODE_RESULTBarcodeClose();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

2.5.3 BarcodeIsOpened

Checking whether the Barcode device is already opened.If BarcodeOpen() was successfully executed, the Barcode will be under open status.

```
BOOLBarcodeIsOpened()
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

2.5.4 BarcodeSetCallback

When scanning has been done completely, register the application program's callback function that called by Barcode system driver.

```
BARCODE_RESULTBarcodeSetCallback (
    BARCODECALLBACK pFunc,
);
```

Parameters

pFunc

callback function

Return Values

BARCODE_RESULT_SUCCESS will be returned if the callback function registered properly.

2.5.5 BarcodeGetVersionInfo

Starting to read Barcode asynchronously.

```
BARCODE_RESULT    BarcodeVersionInfo();
```

Parameters

None

Return Values

This function always return BARCODE_RESULT_UNSUPPORTED

2.5.6 BarcodeStart

Starting to scan Barcode.

BARCODE_RESULTBarcodeStart();

Parameters

None

Return Values

Successfully to start to scanning barcode, BARCODE_RESULT_SUCCESS will be returned.

2.5.7 BarcodeStartScanRawData

Starting to read Barcode asynchronously. The resulf of reading is not string but the arrangement of byte.

BARCODE_RESULT BarcodeStartScanRawData();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

If successfully excuted, reading Barcode will succeed or trying reading Barcode before Stop function is called.

Only in support of the 2D Barcode Scanner.

2.5.8 BarcodeStop

Stop to scan Barcode.

BARCODE_RESULTBarcodeStop();

Parameters

None

Return Values

BARCODE_RESULT_SUCCESS will be returned, if successfully stopped to scanning Barcode.

2.5.9 BarcodeStopScanRawData

Stop readingBarcode.

BARCODE_RESULT BarcodeStopScanRawData();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

2.5.10 BarcodeGetScannedData

Reading barcode information which recognized successfully from AADF.

```
BARCODE_RESULTBarcodeGetScannedData (
    LPWSTRBarcodeValue,
    LPWSTRBarcodeTypeName,
    LPWSTRBarcodeTypeId,
);
```

Parameters

BarcodeValue

pointer of string which will be stored barcode value.

BarcodeTypeName

pointer of string which will be stored barcode type name

BarcodeTypeId

pointer of string which will be stored barcode type id

Return Values

BARCODE_RESULT_SUCCESS will be returned if the scanned barcode pasted properly.

Notes

This function must be called from within registered delegate function by user.

2.5.11 BarcodeGetBarcodeRawData

Reading raw data of recognized Barcode.

```
BARCODE_RESULT   BarcodeGetBarcodeRawData (
    BYTE * BarcodeValue,
    WORD * pLength,
    LPWSTR BarcodeTypeName,
    LPWSTR BarcodeTypeId,
);
```

Parameters

BarcodeValue

Pointer of the arrangement of byte in which raw data of Barcode will be stored

pLength

pointer of variables in which the length of raw data will be stored

BarcodeTypeName

Barcode Type Name

BarcodeTypeId

Barcode Type ID

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

This function must be called from: within Callback function or in receiving WM_BARCODE_ONSCANED messages.

2.5.12 BarcodeInitCapture

In order to capture images through 2D Barcode, assigned necessary system resource and initialize related parameters.

BARCODE_RESULTBarcodeInitCapture ();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner

2.5.13 BarcodeDeinitCapture

In order to capture image, clear the assigned system resource.

BARCODE_RESULTBarcodeDeinitCapture ();

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner

2.5.14 BarcodeSetPreviewHwnd

Used in 2D Barcode restrictively. In 2D Barcode device, if the application would like to see preview window images through 2D BARCODE, transmit windows handle of preview window screen.

```
BARCODE_RESULTBarcodeSetPreviewHwnd (
    HWNDhWnd,
);
```

Parameters

hWnd

preview window screen's windows handle

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner.

2.5.15 BarcodeStartPreview

Drawing the image data that transmitted from Barcode to preview window screen.

```
BARCODE_RESULTBarcodeStartPreview ();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner.

2.5.16 BarcodeStopPreview

Stop to preview window.

```
BARCODE_RESULTBarcodeStopPreview ();
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner

2.5.17 BarcodeDoCapture

Capture image data and store into files that through 2D Barcode scanner.

```
BARCODE_RESULTBarcodeDoCapture (
    BARCODECAPTUREPARAMS*pCaputreParams
);
```

Parameters

pCaputreParams

transmit image's resolution, format, and storage path.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

Only in support of 2D Barcode Scanner.

2.5.18 BarcodeGetEnableStateAll

Returning Enable/Disable condition about all Symbologies.

```
BARCODE_RESULTBarcodeGetEnableStateAll (
    BOOL * bSymbolTable
);
```

Parameters

pSymbolTable

Pointer of arrangement of BOOL in which Enable/Disable condition will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES, in case of OCR, due to the property of Symbology, it is so difficult to check condition that directly checking setting value(GetSymbologyConfig) through reading is recommended.

2.5.19 BarcodeEnableSymbologiesAll

Setting Enable/Disable condition about all Symbologies.

```
BARCODE_RESULTBarcodeGetEnableStateAll (
    BOOL bEnable
);
```

Parameters

bEnable

모든 symbology에적용할 Enable/Disable 상태

True: 모든 symbology가 Enable 됨

False: 모든 symbology가 Disable 됨

Enable/ Disable condition to be applied to all Symbologies

True: all symbology enabled

False:all symbology disabled

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned

Notes

2.5.20 BarcodeEnableSymbologies

Enabling the specific Symbology.

Enablign

```
BARCODE_RESULTBarcodeEnableSymbologies (
    BOOL *pSymbolTable
);
```

Parameters

pSymbolTable

pointer of arrangement of BOOL in which information of symbology to be enabled is stored.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES. Set the symbology to be enabled to "true" using value defined in SYMBOLOGIES_1D or SYMBOLOGIES_2DSWD.

2.5.21 BarcodeDisableSymbologies

Disabling the specific Symbology

```
BARCODE_RESULTBarcodeDisableSymbologies (
    BOOL *pSymbolTable
);
```

Parameters

pSymbolTable

pointer of arrangement of BOOL in which information of symbology to be disabled is stored.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

The size of pSymbolTable must be set to NUM_OF_1D_SYMBOLOGIES or NUM_OF_2DSWD_SYMBOLOGIES. Set the symbology to be enabled to "true" using value defined in SYMBOLOGIES_1D or SYMBOLOGIES_2DSWD.

2.5.22 BarcodeGetSymbologyConfig

Getting the set value of specific Symbology from Scanner.

```
BARCODE_RESULTBarcodeGetSymbologyConfig (
    BARCODE_SYMBOL_CONFIG *pSymbolConfig
);
```

Parameters

pSymbolConfig

structure where the set value of Symbology imported from Scanner will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Reference to 1.2.3 SYMBOL_CONFIG

2.5.23 BarcodeSetSymbologyConfig

Storing the set value of specific Symbology in Scanner.

```
BARCODE_RESULTBarcodeSetSymbologyConfig (
    BARCODE_SYMBOL_CONFIG *pSymbolConfig
);
```

Parameters

pSymbolConfig

The structure where the set value of Symbology to be stored in Scanner is stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Reference to 2.3.3 BARCODE_SYMBOL_CONFIG

2.5.24 BarcodeGetGeneralConfig

Getting set value of 1D Barcode Scanner like security level, scanning angle.

```
BARCODE_RESULTBarcodeGetGeneralConfig (
    BARCODE_GENERAL_CONFIG *pGeneralConfig
);
```

Parameters

pGeneralConfig

structure in which set value imported from Scanner will be stored

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of 1D Barcode scanner.

Reference to 2.3.2 GENERAL_CONFIG

2.5.25 BarcodeSetGeneralConfig

Setting value of 1D Barcode Scanner like security level, scanning angle

```
BARCODE_RESULTBarcodeSetGeneralConfig (
    BARCODE_GENERAL_CONFIG *pGeneralConfig
);
```

Parameters

pGeneralConfig

The structure in which set value that will be stored in Scanner is Stored.

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of 1D Barcode scanner.

Reference to 2.3.2 GENERAL_CONFIG

2.5.26 BarcodeSetDefaultSymbol

Resetting all set values of Symbology

```
BARCODE_RESULTBarcodeSetDefaultSymbol ( );
```

Parameters

None

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

2.5.27 BarcodeSetScanningLightMode

Setting LED light(aimers, illumination) mode while 2D Barcode Scanner scanning.

```
BARCODE_RESULT   BarcodeSetScanningLightMode (
    int nMode
);
```

Parameters

nMode

0 : Neither aimers nor illumination

1 : Illumination only

2 : Aimers only

3 : Both aimers and illumination

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

2.5.28 BarcodeSetLeaveLightsOn

Setting on/off switch value of LED light(aimers, illumination) while 2D Barcode Scanner scanning.

```
BARCODE_RESULT   BarcodeSetLeaveLightsOn (
    BOOLbEnable
);
```

Parameters

bEnable

true: LED light on while scanning.

False: LED light on and off while scanning

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

2.5.29 BarcodeGetLeaveLightsOn

Getting on/off value of LED light(aimers, illumination) while 2D Barcode Scanner scanning.

```
BARCODE_RESULT    BarcodeGetLeaveLightsOn (
        BOOL *pbEnable
);
```

Parameters

pbEnable

true: LED light on while scanning.

False: LED light on and off while scanning

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

2.5.30 BarcodeGetIlluminationPowerLevel

Get the Illumination power level of 2D Barcode Scanner.

```
BARCODE_RESULT    BarcodeGetIlluminationPowerLevel (
        BYTE* pPwrLevel
);
```

Parameters

pPwrLevel

illumination power level

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of Motorola SE4500 Scanner.

2.5.31 BarcodeSetIlluminationPowerLevel

Set the Illumination power level of 2D Barcode Scanner.

```
BARCODE_RESULT    BarcodeSetIlluminationPowerLevel (
        BYTE PwrLevel
);
```

Parameters

PwrLevel

illumination power level

Return Values

If successfully executed, BARCODE_RESULT_SUCCESS will be returned.

Notes

Only in support of Motorola SE4500 Scanner.